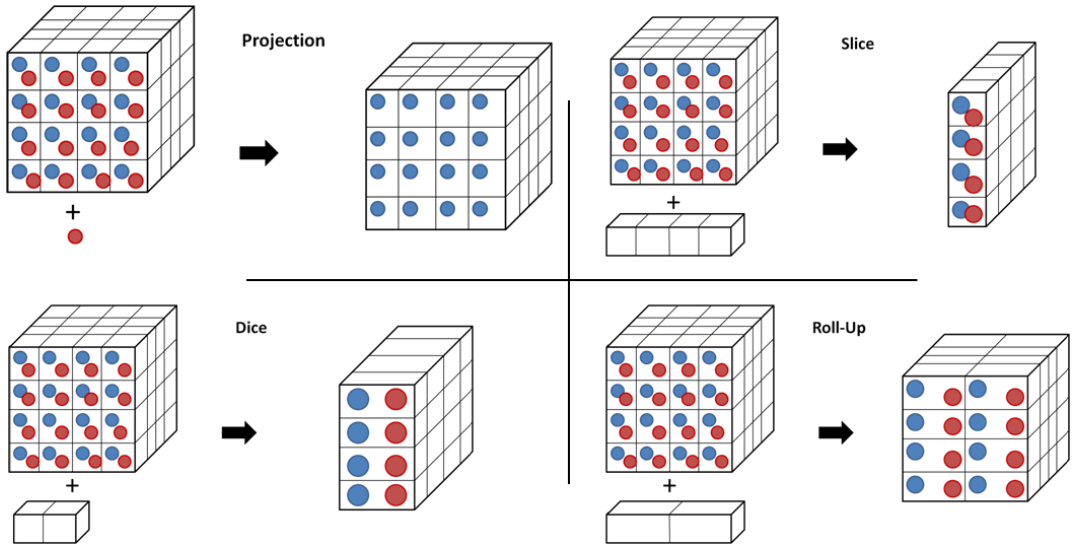
Interacting with Statistical Linked Data via OLAP Operations

Benedikt Kämpgen, Seán O'Riain, Andreas Harth

International Workshop on Interacting with Linked Data, Extended Semantic Web Conference 2012

28 May 2012

Institute of Applied Informatics and Formal Description Methods (AIFB)



Outline

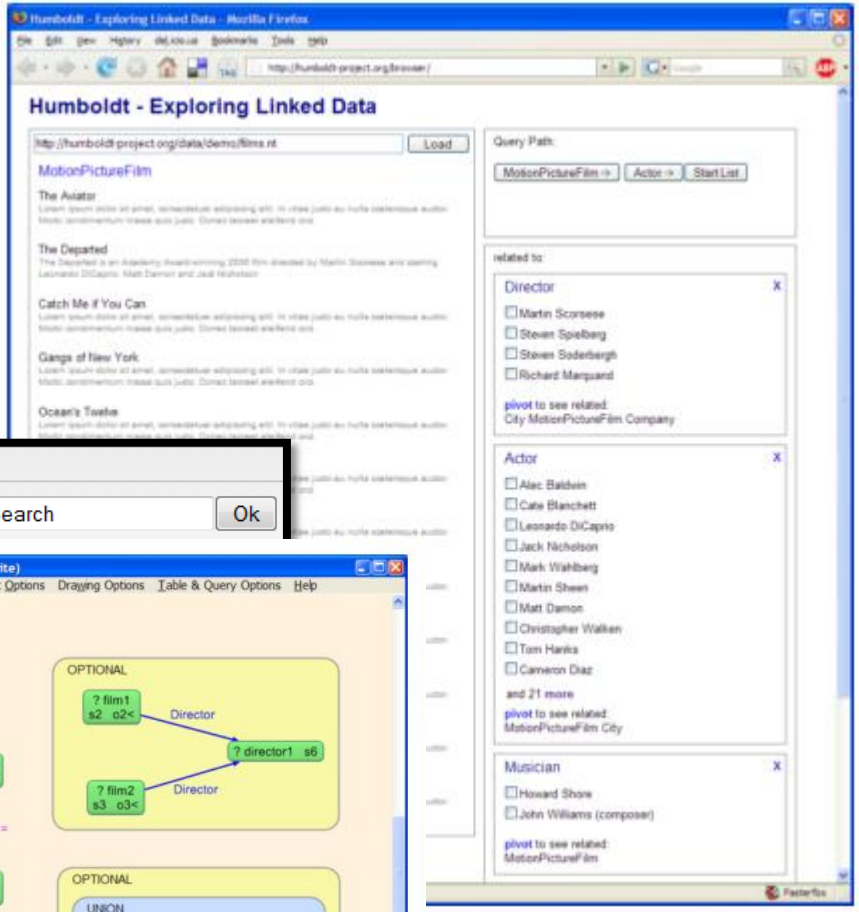
- Interacting with Statistical Linked Data
- Data Cubes as RDF
- OLAP Operations and Queries
- OLAP Queries in SPARQL
- Evaluation/Discussion/Conclusions

Outline

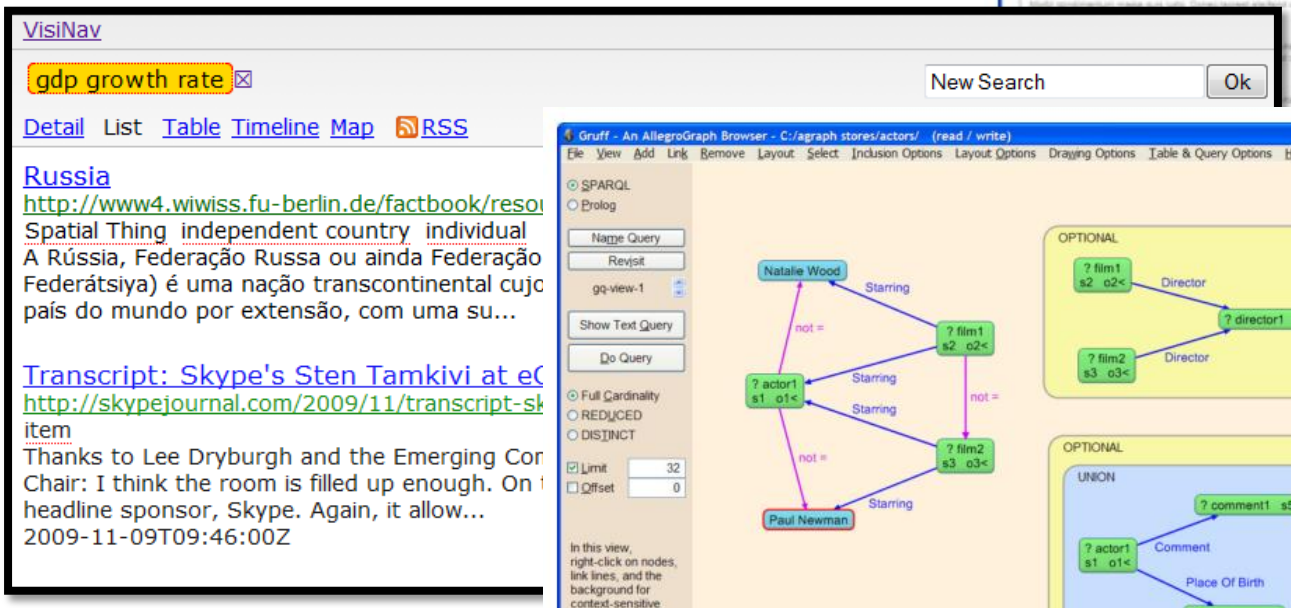
- **Interacting with Statistical Linked Data**
- Data Cubes as RDF
- OLAP Operations and Queries
- OLAP Queries in SPARQL
- Evaluation/Discussion/Conclusions

Decision Support using Statistical Linked Data

Difficulties with statistics as Linked Data:
No **aggregation**, **spreadsheets and pivot tables**, or **drill-down**.



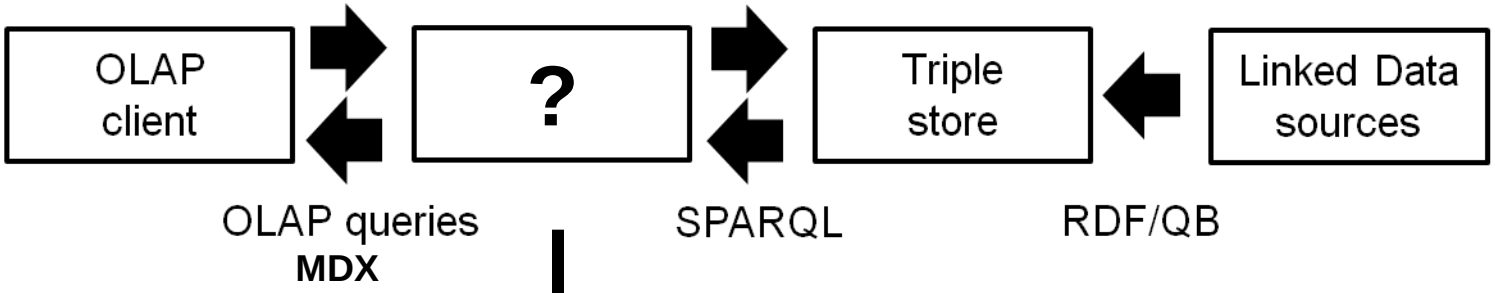
Humboldt faceted search [3]



Visinav Linked Data browser [1]

Gruff SPARQL query builder [2]

Online Analytical Processing of Statistical Linked Data



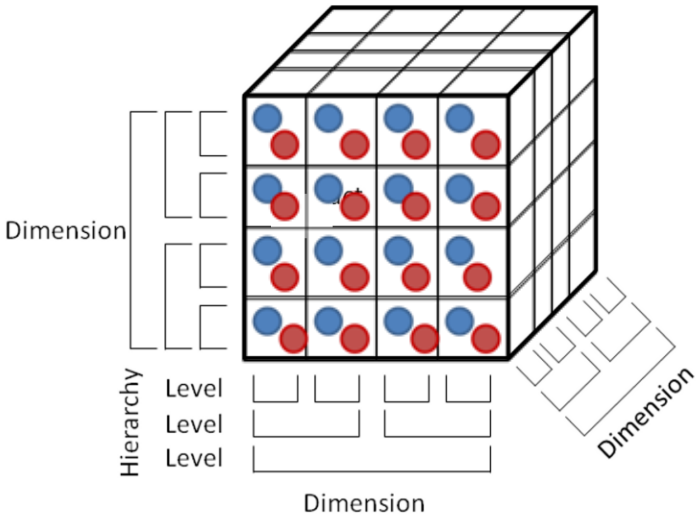
- Typical OLAP client
- OLAP operations
- OLAP queries
- Multidimensional Expression Language (MDX)

- Statistical Linked Data re-using the RDF Data Cube Vocabulary (QB)
- Crawled into triple store
- SPARQL endpoint

Problem: How to map MDX queries to SPARQL queries?

Scenario

Financial data about US companies as RDF in triple store



Data cube: Financial disclosures

Measures: Cost of goods sold, Sales (UDF aggr. funct.)

Dimensions: Issuer, Dtstart, Dtend, Segment

OLAP query

http://diveintechno.com/information-technology-business-analyst.html/

Business analyst with OLAP client

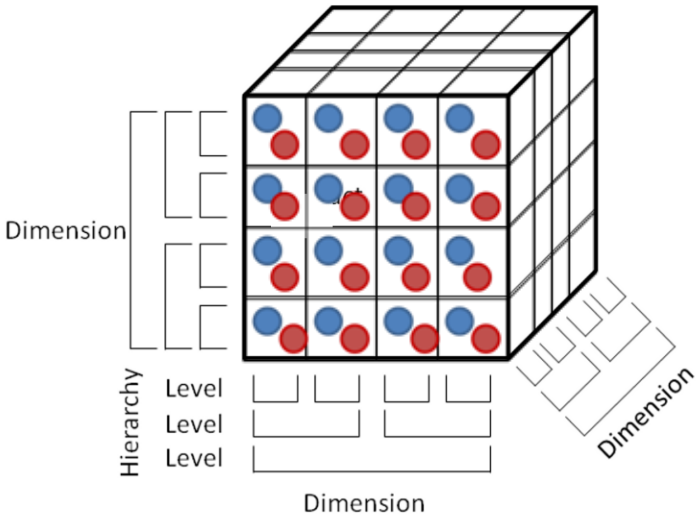
Pivot table

Columns (issuer) / Rows (dtstart, dtend)		RAYONIER INC	WEYERHAEUSER CO
2009-01-01	2009-3-31	*1,100,335 USD*	*0 values*
2009-04-01	2009-06-30	*2 values*	*2,300,800 USD*
...	...	...	...

Filters: CostOfGoodsSold

Scenario

Financial data about US companies as RDF in triple store



Data cube: Financial disclosures

Measures: Cost of goods sold, Sales (UDF aggr. funct.)

Dimensions: Issuer, Dtstart, Dtend, Segment

OLAP query

http://diveintechno.com/information-technology-business-analyst.html/

Business analyst with OLAP client

Pivot table

Columns (issuer) / Rows (dtstart, dtend)		RAYONIER INC	WEYERHAEUSER CO
2009-01-01	2009-3-31	*1,100,335 USD*	*0 values*
2009-04-01	2009-06-30	*2 values*	*2,300,800 USD*
...	...	...	...

Filters: CostOfGoodsSold

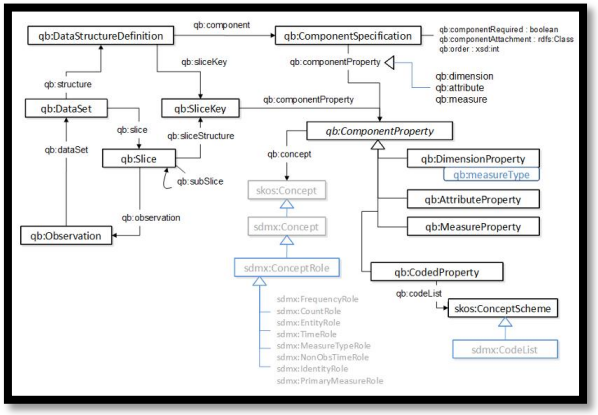
Approach - Overview

1. **Data cubes** mapped to RDF re-using QB
2. **OLAP operations** with data cubes as input/output
3. **OLAP queries** as sets of OLAP operations on data cubes
4. **SPARQL queries** from OLAP queries

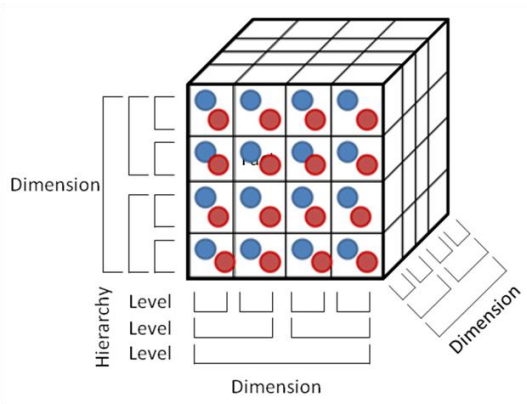
Outline

- Interacting with Statistical Linked Data
- **Data Cubes as RDF**
- OLAP Operations and Queries
- OLAP Queries in SPARQL
- Evaluation/Discussion/Conclusions

Data Cubes as RDF (1)



The RDF Data Cube Vocabulary [4]



Common Data Cube Model

DataCubeSchema defines the set of data cube schemas as $\{(?x, D, M) \in (\mathcal{I} \cup \mathcal{B}) \times \mathcal{D} \times \mathcal{M} | (?x \text{ a } qb{:}DataStructureDefinition \wedge \forall d \in D(?x\ qb{:}componentProperty\ ?comp \wedge ?comp\ qb{:}dimensionProperty\ iri(d)) \wedge \forall m \in M(?x\ qb{:}componentProperty\ ?comp \wedge ?comp\ qb{:}measureProperty\ iri(m)))\}$.

DataCube defines the set of data cubes as $DataCube = \{(cs, F) \in DataCubeSchema \times \mathcal{F} | cs = (?x, D, M) \wedge D = \{D_0, \ldots, D_{|D|}\} \wedge M = \{m_0, \ldots, m_{|M|}\} \wedge \forall c \in F(c = (?obs, c_0, \ldots, c_{|D|}, e_0, \ldots, e_{|M|}) : (?obs\ qb{:}dataSet\ ?ds \wedge ?ds\ qb{:}structure\ ?x) \wedge \forall D_i \in D(?obs\ iri(D_i)\ c_i \wedge iri(D_i)\ qb{:}codeList\ ?h \wedge ?l\ skos{:}inScheme\ ?h \wedge ?v\ skos{:}member\ ?l \wedge (?v\ skos{:}notation\ ?c_i \vee ?v\ skos{:}exactMatch\ ?c_i)) \wedge \forall m_i \in M(?obs\ iri(m_i)\ e_i \vee e_i = null)\}$.

Data Cubes as RDF (2)

Scenario

- Cube URIs:
  - `edgar:SecCubeGrossProfitMargin`
- Dimension URIs:
  - `edgar:issuer`
  - `edgar:dtstart`
  - `edgar:dtend`
  - `edgar:segment`
- Measure URIs:
  - `edgar:CostOfGoodsSold`
  - `edgar:Sales`

Materialising Data Cubes

- Computing all possible facts of a data cube (materialisation) requires $2^{|D|}$ subqueries [5].

Issuer	Dtstart	Dtend	Segment	COGS	Sales
RAYONIER INC	2009-01-01	2009-03-31	Business	1,100,335 USD	2,000,000
RAYONIER INC	2009-04-01	2009-06-30	Business	500,442 USD	1,245,000
…	…	…	…	…	…
RAYONIER INC	2009-01-01	2009-03-31	**ALL**	1,100,335 USD	2,000,000
…	…	…	…	…	…
…	…	**ALL**	…	…	…
…	…	…	…	…	…
…	**ALL**	…	…	…	…
…	…	…	…	…	…
ALL	…	…	…	…	…
…	…	…	…	…	…
…	…	**ALL**	**ALL**	…	…
…	…	…	…	…	…
ALL	**ALL**	**ALL**	**ALL**	…	…

Scenario: Financial disclosure: Issuer (625), Dtstart (27), Dtend (20), Segment (21,227) has **7,813,772** possible facts.

Materialising Data Cubes

- Computing all possible facts of a data cube (materialisation) requires $2^{|D|}$ subqueries [5].

Issuer	Dtstart	Dtend	Segment	COGS	Sales
RAYONIER INC	2009-01-01	2009-03-31	Business	1,100,335 USD	2,000,000
RAYONIER INC	2009-04-01	2009-06-30	Business	500,442 USD	1,245,000
…	…	…	…	…	…
RAYONIER INC	2009-01-01	2009-03-31	**ALL**	1,100,335 USD	2,000,000
…	…	…	…	…	…
…	…	**ALL**	…	…	…
…	…	…	…	…	…
…	**ALL**	…	…	…	…
…	…	…	…	…	…
ALL	…	…	…	…	…
…	…	…	…	…	…
…	…	**ALL**	**ALL**	…	…
…	…	…	…	…	…
ALL	**ALL**	**ALL**	**ALL**	…	…

Scenario: Financial disclosure: Issuer (625), Dtstart (27), Dtend (20), Segment (21,227) has **7,813,772** possible facts.

Materialising Data Cubes

- Computing all possible facts of a data cube (materialisation) requires $2^{|D|}$ subqueries [5].

Issuer	Dtstart	Dtend	Segment	COGS	Sales
RAYONIER INC	2009-01-01	2009-03-31	Business	1,100,335 USD	2,000,000
RAYONIER INC	2009-04-01	2009-06-30	Business	500,442 USD	1,245,000
…	…	…	…	…	…
RAYONIER INC	2009-01-01	2009-03-31	**ALL**	1,100,335 USD	2,000,000
…	…	…	…	…	…
…	…	**ALL**	…	…	…
…	…	…	…	…	…
…	**ALL**	…	…	…	…
…	…	…	…	…	…
ALL	…	…	…	…	…
…	…	…	…	…	…
…	…	**ALL**	**ALL**	…	…
…	…	…	…	…	…
ALL	**ALL**	**ALL**	**ALL**	…	…

Scenario: Financial disclosure: Issuer (625), Dtstart (27), Dtend (20), Segment (21,227) has **7,813,772** possible facts.

Materialising Data Cubes

- Computing all possible facts of a data cube (materialisation) requires $2^{|D|}$ subqueries [5].

Issuer	Dtstart	Dtend	Segment	COGS	Sales
RAYONIER INC	2009-01-01	2009-03-31	Business	1,100,335 USD	2,000,000
RAYONIER INC	2009-04-01	2009-06-30	Business	500,442 USD	1,245,000
…	…	…	…	…	…
RAYONIER INC	2009-01-01	2009-03-31	**ALL**	1,100,335 USD	2,000,000
…	…	…	…	…	…
…	…	**ALL**	…	…	…
…	…	…	…	…	…
…	**ALL**	…	…	…	…
…	…	…	…	…	…
ALL	…	…	…	…	…
…	…	…	…	…	…
…	…	**ALL**	**ALL**	…	…
…	…	…	…	…	…
ALL	**ALL**	**ALL**	**ALL**	…	…

Scenario: Financial disclosure: Issuer (625), Dtstart (27), Dtend (20), Segment (21,227) has **7,813,772** possible facts.

Materialising Data Cubes

- Computing all possible facts of a data cube (materialisation) requires $2^{|D|}$ subqueries [5].

Issuer	Dtstart	Dtend	Segment	COGS	Sales
RAYONIER INC	2009-01-01	2009-03-31	Business	1,100,335 USD	2,000,000
RAYONIER INC	2009-04-01	2009-06-30	Business	500,442 USD	1,245,000
…	…	…	…	…	…
RAYONIER INC	2009-01-01	2009-03-31	**ALL**	1,100,335 USD	2,000,000
…	…	…	…	…	…
…	…	**ALL**	…	…	…
…	…	…	…	…	…
…	**ALL**	…	…	…	…
…	…	…	…	…	…
ALL	…	…	…	…	…
…	…	…	…	…	…
…	…	**ALL**	**ALL**	…	…
…	…	…	…	…	…
ALL	**ALL**	**ALL**	**ALL**	…	…

Scenario: Financial disclosure: Issuer (625), Dtstart (27), Dtend (20), Segment (21,227) has **7,813,772** possible facts.

Materialising Data Cubes

- Computing all possible facts of a data cube (materialisation) requires $2^{|D|}$ subqueries [5].

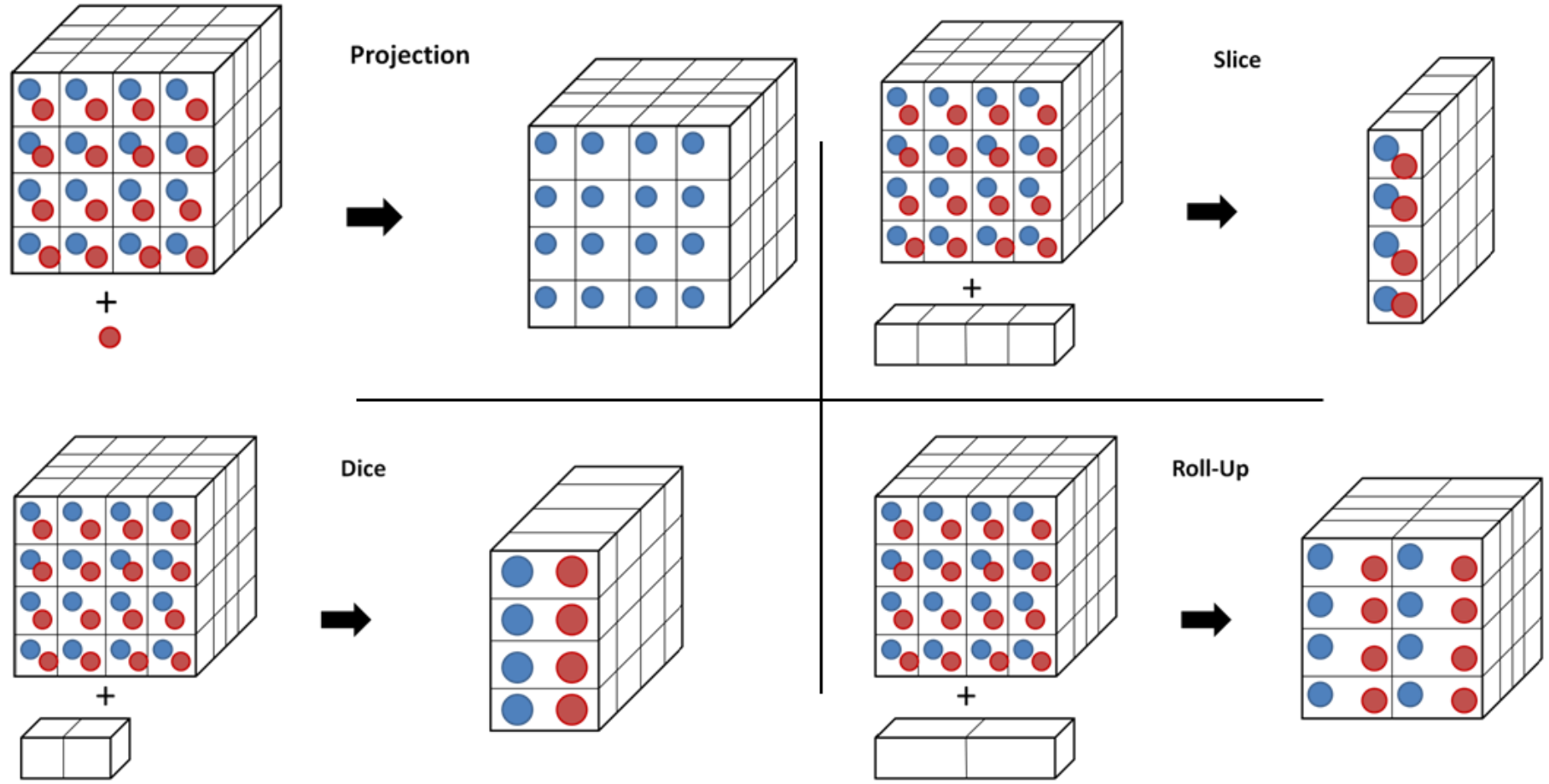
Issuer	Dtstart	Dtend	Segment	COGS	Sales
RAYONIER INC	2009-01-01	2009-03-31	Business	1,100,335 USD	2,000,000
RAYONIER INC	2009-04-01	2009-06-30	Business	500,442 USD	1,245,000
…	…	…	…	…	…
RAYONIER INC	2009-01-01	2009-03-31	**ALL**	1,100,335 USD	2,000,000
…	…	…	…	…	…
…	…	**ALL**	…	…	…
…	…	…	…	…	…
…	**ALL**	…	…	…	…
…	…	…	…	…	…
ALL	…	…	…	…	…
…	…	…	…	…	…
…	…	**ALL**	**ALL**	…	…
…	…	…	…	…	…
ALL	**ALL**	**ALL**	**ALL**	…	…

Scenario: Financial disclosure: Issuer (625), Dtstart (27), Dtend (20), Segment (21,227) has **7,813,772** possible facts.

Outline

- Interacting with Statistical Linked Data
- Data Cubes as RDF
- **OLAP Operations and Queries**
- OLAP Queries in SPARQL
- Evaluation/Discussion/Conclusions

OLAP Operations – Informal



OLAP Operations – Formal

Projection is defined as $Projection : DataCube \times Measure \rightarrow DataCube$ and removes a measure from the input cube and allows to query only for specific measures. We evaluate *Projection* by removing a measure from the subcube query tuple.

Slice is defined as $Slice : DataCube \times Dimension \rightarrow DataCube$ and removes a dimension from the input cube and aggregates over the members of a dimension. We evaluate *Slice* by setting the tuple element of that dimension to ALL.

Dice, Roll-Up (see paper)

- Input/Output: data cubes; OLAP operations can be nested.
- Nested set of OLAP operations on data cube lead to OLAP query.
- OLAP query returns subset of facts in materialised data cube.

Scenario

```
OLAP operations:
1) Q = Projection(edgar:SecCubeGrossProfitMargin, ed:Sales)
2) Q = Slice(Q, ed:Segment)
```

Issuer	Dtstart	Dtend	Segment	COGS
RAYONIER INC	2009-01-01	2009-03-31	**ALL**	**1,100,335 USD**
RAYONIER INC	2009-04-01	2009-06-30	**ALL**	**500,442 USD**
…	…	…	**…**	**…**
RAYONIER INC	2009-01-01	2009-03-31	**ALL**	**1,100,335 USD**
…	…	…	**…**	**…**

OLAP Queries

OLAP query (subcube query)

$(q_0, ..., q_{|D|}, m_0, ..., m_{|M|})$ *with* $dom(q_i) = \{?, ALL, x\}$ *with* ? *for an* inquired *dimension, with* ALL *for an aggregated dimension, with* x *for one or more members to fix a dimension, and with* m_i *a measure to query for.*

Scenario

```
(?,?,?,?,COGS, Sales)
```

Full-cube query: All explicitly contained facts in a data cube.

```
({RAYONIER INC},
{2009-01-01},{2009-03-31},
{Business},COGS,Sales)
```

Point query: One specific explicitly contained fact from a data cube.

```
(ALL,ALL,ALL,ALL,COGS,Sales)
```

Fully-aggregated query: Implicitly contained fact aggregating over all dimensions.

OLAP Queries as Sets of OLAP operations on a Data Cube

Projection is defined as $Projection : DataCube \times Measure \rightarrow DataCube$ and removes a measure from the input cube and allows to query only for specific measures. We evaluate *Projection* by removing a measure from the subcube query tuple.

Slice is defined as $Slice : DataCube \times Dimension \rightarrow DataCube$ and removes a dimension from the input cube and aggregates over the members of a dimension. We evaluate *Slice* by setting the tuple element of that dimension to *ALL*.

Dice, Roll-Up (see paper)

Scenario

```
OLAP operations:
1) Q = Projection(edgar:SecCubeGrossProfitMargin, edgar:Sales)
2) Q = Slice(Q, edgar:Segment)
```

```
OLAP query:
Init) Q = (?,?,?,?,Cost of goods sold, Sales)
1) Q = (?,?,?,?,Cost of goods sold)
2) Q = (?,?,?,ALL,Cost of goods sold)
```

Outline

- Interacting with Statistical Linked Data
- Data Cubes as RDF
- OLAP Operations and Queries
- **OLAP Queries in SPARQL**
- Evaluation/Discussion/Conclusions

OLAP Queries in SPARQL

- Given a **subcube query**, resulting from a set of nested OLAP operations on a data cube:

$$Q = (q_0, \ldots, q_{|D|}, m_0, ..., m_{|M|})$$

- Translated into SPARQL query:
  1. Initial SPARQL query for all facts from the data cube.
  2. For each selected measure, add aggregated selection and graph pattern.
  3. For each inquired dimension, add selection, graph patterns, and group by.
  4. For each fixed dimension, add graph pattern.

Scenario

1.

1. Initial SPARQL query for all facts from the data cube.

```
select ?dimMem0 ?dimMem1 ?dimMem2 count(xsd:decimal(?
    measureValue0)) sum(xsd:decimal(?measureValue0))
where {
?obs qb:dataSet ?ds.
?ds qb:structure edgar:SecCubeGrossProfitMargin.
?obs edgar:issuer ?values0.
        ?dimMem0 skos:member ?level0.
        ?level0 skos:inScheme ?hierarchy0.
        edgar:issuer qb:codeList ?hierarchy0.
        ?dimMem0 skos:exactMatch ?values0.
?obs edgar:dtstart ?values1.
        ?dimMem1 skos:member ?level1.
        ?level1 skos:inScheme ?hierarchy1.
        edgar:dtstart qb:codeList ?hierarchy1.
        ?dimMem1 skos:notation ?values1.
?obs edgar:dtend ?values2.
        ?dimMem2 skos:member ?level2.
        ?level2 skos:inScheme ?hierarchy2.
        edgar:dtend qb:codeList ?hierarchy2.
        ?dimMem2 skos:notation ?values2.
?obs edgar:CostOfGoodsSold ?measureValue0.
} group by ?dimMem0 ?dimMem1 ?dimMem2
```

SPARQL query translated from subcube query

Scenario

2. For each selected measure, add aggregated selection and graph pattern.

2.1

```
select ?dimMem0 ?dimMem1 ?dimMem2 count(xsd:decimal(?
    measureValue0)) sum(xsd:decimal(?measureValue0))
where {
?obs qb:dataSet ?ds.
?ds qb:structure edgar:SecCubeGrossProfitMargin.
?obs edgar:issuer ?values0.
        ?dimMem0 skos:member ?level0.
        ?level0 skos:inScheme ?hierarchy0.
        edgar:issuer qb:codeList ?hierarchy0.
        ?dimMem0 skos:exactMatch ?values0.
?obs edgar:dtstart ?values1.
        ?dimMem1 skos:member ?level1.
        ?level1 skos:inScheme ?hierarchy1.
        edgar:dtstart qb:codeList ?hierarchy1.
        ?dimMem1 skos:notation ?values1.
?obs edgar:dtend ?values2.
        ?dimMem2 skos:member ?level2.
        ?level2 skos:inScheme ?hierarchy2.
        edgar:dtend qb:codeList ?hierarchy2.
        ?dimMem2 skos:notation ?values2.
?obs edgar:CostOfGoodsSold ?measureValue0.
} group by ?dimMem0 ?dimMem1 ?dimMem2
```

2.2

SPARQL query translated from subcube query

Scenario

3. For each inquired dimension, add selection, graph patterns, and group by.

3.1

```
select ?dimMem0 ?dimMem1 ?dimMem2 count(xsd:decimal(?
    measureValue0)) sum(xsd:decimal(?measureValue0))
where {
?obs qb:dataSet ?ds.
?ds qb:structure edgar:SecCubeGrossProfitMargin.
```

3.2

```
?obs edgar:issuer ?values0.
        ?dimMem0 skos:member ?level0.
        ?level0 skos:inScheme ?hierarchy0.
        edgar:issuer qb:codeList ?hierarchy0.
        ?dimMem0 skos:exactMatch ?values0.
?obs edgar:dtstart ?values1.
        ?dimMem1 skos:member ?level1.
        ?level1 skos:inScheme ?hierarchy1.
        edgar:dtstart qb:codeList ?hierarchy1.
        ?dimMem1 skos:notation ?values1.
?obs edgar:dtend ?values2.
        ?dimMem2 skos:member ?level2.
        ?level2 skos:inScheme ?hierarchy2.
        edgar:dtend qb:codeList ?hierarchy2.
        ?dimMem2 skos:notation ?values2.
?obs edgar:CostOfGoodsSold ?measureValue0.
```

3.3

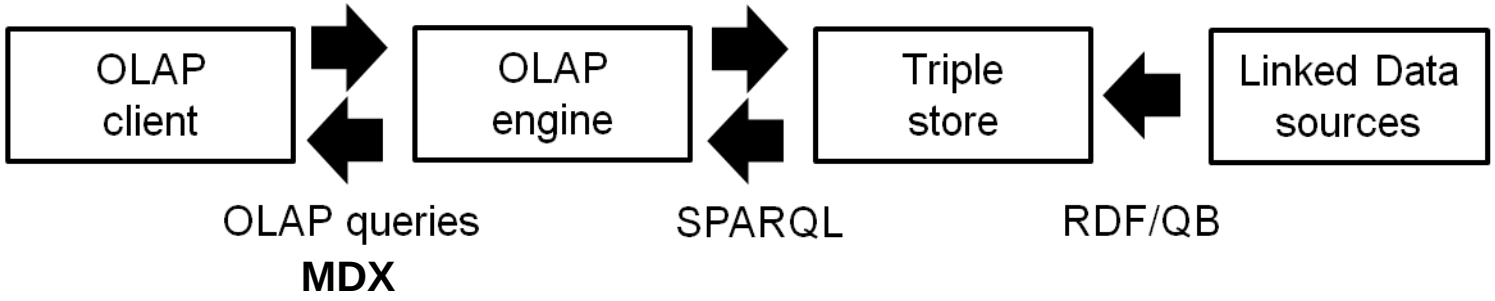
```
} group by ?dimMem0 ?dimMem1 ?dimMem2
```

SPARQL query translated from subcube query

Outline

- Interacting with Statistical Linked Data
- Data Cubes as RDF
- OLAP Operations and Queries
- OLAP Queries in SPARQL
- **Evaluation/Discussion/Conclusions**

Evaluation/Discussion



Scenario

- OLAP query took 18sec and returned 58 facts to be filled into the requested pivot table.

OLAP query could be faster

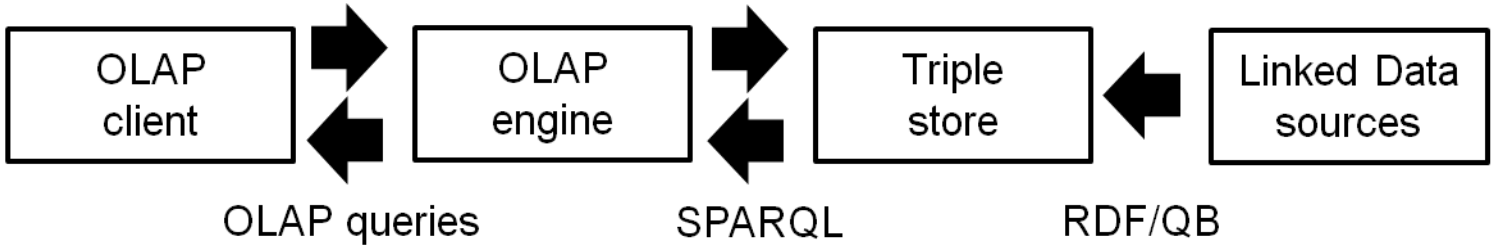
- Data from the data cube is queried on demand, and no materialisation is done.
- No dice without slice; uneccessary computation of facts.
- No direct correspondence of SPARQL result and requested pivot table: Populating pivot table requires join.

Conclusions

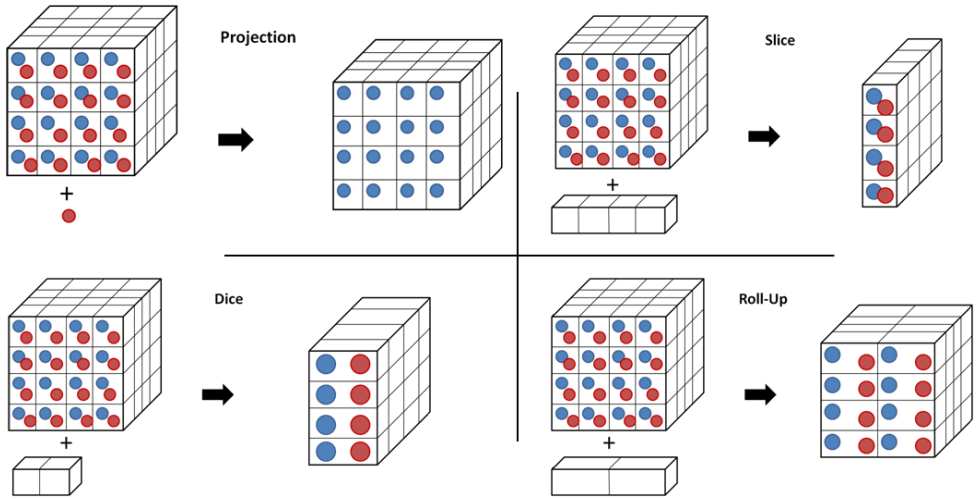
- First work on OLAP-to-SPARQL mapping

- Future work
  - Investigating intuitive user interactions for navigating Data Cubes as Linked Data.
  - Considering OLAP hierarchies and multi-cube operations (e.g., Drill-Across).
  - Implementing an OLAP engine (olap4ld) for optimising OLAP query execution.

Thanks!



Feedback?
Questions?



More about OLAP of Linked Data: http://semanticweb.org/wiki/OLAP_of_Linked_Data

References

[1] Harth A. VisiNav : Visual Web Data Search and Navigation. In: *DEXA 2009*.; 2009:214-228.

[2] Aasman J, Cheetham K. RDF Browser for Data Discovery and Visual Query Building. *Interfaces*. 2011:1-4.

[3] Kobilarov G, Dickinson I. Humboldt : Exploring Linked Data. *LDOW*. 2008.

[4] The RDF Data Cube Vocabulary: http://www.w3.org/TR/vocab-data-cube/

[5] Gray, J., Bosworth, a., Lyaman, a., Pirahesh, H.: Data cube: a relational aggregation operator generalizing GROUP-BY, CROSS-TAB, and SUB-TOTALS. Proceedings of the Twelfth International Conference on Data Engineering pp. 152-159 (1995)

Backup: Multidimensional Model in RDF

Member defines the set of members as $Member = \{?x \in (\mathcal{I} \cup \mathcal{B}) | (?x \text{ a } skos{:}Concept)\}$. Let $\mathcal{V} = 2^{Member}$, $V \in \mathcal{V}$, $ROLLUPMEMBER \subseteq Member \times Member$, $rollupmember(V) = \{(v_1, v_2) \in V \times V | (iri(v_1)\ skos{:}broader\ iri(v_2) \vee iri(v_2)\ skos{:}narrower\ iri(v_1))\}$

Level defines the set of levels as $Level = \{(?x, V, rollupmember(V)) \in (\mathcal{I} \cup \mathcal{B}) \times \mathcal{V} \times ROLLUPMEMBER | (?x \text{ a } skosclass{:}ClassificationLevel \wedge \forall v \in V(iri(v)\ skos{:}member\ ?x)\}$. Let $\mathcal{L} = 2^{Level}$, $L \in \mathcal{L}$, $ROLLUPLEVEL \subseteq Level \times Level$, $rolluplevel(L) = \{(l_1, l_2) \in L \times L | (iri(l_1)\ skosclass{:}depth\ x) \wedge (iri(l_2)\ skosclass{:}depth\ y) \wedge x \leq y))\}$

Hierarchy defines the set of hierarchies as $Hierarchy = \{(?x, L, rolluplevel(L)) \in (\mathcal{I} \cup \mathcal{B}) \times \mathcal{L} \times ROLLUPLEVEL | (?x \text{ a } skos{:}ConceptScheme\ ) \wedge \forall l \in L(iri(l)\ skos{:}inScheme\ ?x)\}$. Let $\mathcal{H} = 2^{Hierarchy}$.

Dimension defines the set of dimensions as $Dimension = \{(?x, H) \in (\mathcal{I} \cup \mathcal{B}) \times \mathcal{H} | (?x \text{ a } qb{:}DimensionProperty\ ) \wedge \forall h \in H(?x\ qb{:}codeList\ iri(h))\}$. Let $\mathcal{D} = 2^{Dimension}$.

Measure defines the set of measures as $Measure = \{(?x, aggr) \in (\mathcal{I} \cup \mathcal{B}) \times \{UDF\} | (?x \text{ a } qb{:}MeasureProperty\ )\}$ with UDF a default aggregation function since QB so far does not provide a standard way to represent typical aggregation functions such as SUM, AVG and COUNT: if only one value is given, the value itself else the number of values is returned. Conceptually, measures are treated as members of a dimension-hierarchy-level combination labelled "Measures". Let $\mathcal{M} = 2^{Measure}$.

Backup: Dice = Slice + Filter

- Given the query model, a dice + slice cannot be represented, therefore dice = slice + filter.

Scenario

```
Q = (?,?,?,ALL,Cost of goods sold) ≠
({RAYONIER INC, WEYERHAUSER CO},?,?,ALL,Cost
of goods sold)
```

Columns (issuer) / Rows (dtstart, dtend)		RAYONIER INC	WEYERHAEUSER CO	...	...
2009-01-01	2009-3-31	*1,100,335 USD*	*0 values*	...	...
2009-04-01	2009-06-30	*2 values*	*2,300,800 USD*	...	...
...	...	...	...	...	...

Filters: CostOfGoodsSold